UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Conditional Adversarial Networks for Blind Image Deblurring

Author:
Orest KUPYN

Supervisor:
Dr. Rostyslav HRYNIV

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2017

Declaration of Authorship

I, Orest KUPYN, declare that this thesis titled, “Conditional Adversarial Networks for Blind Image Deblurring” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Abstract

Orest KUPYN

Conditional Adversarial Networks for Blind Image Deblurring

We present an end-to-end learning approach for motion deblurring, which is based on conditional GAN and content loss – DeblurGAN. DeblurGAN achieves state-of-the art in structural similarity measure and by visual appearance. The quality of the deblurring model is also evaluated in a novel way on a real-world problem – object detection on (de-)blurred images. The method is 5 times faster than the closest competitor.

Second, we present a novel method of generating synthetic motion blurred images from the sharp ones, which allows realistic dataset augmentation.

Acknowledgements

I would like to express gratitude to the ELEKS Ltd. and ARVI Lab for providing computational resources and to Dmytro Mishkin for supervising this research project and providing valuable feedback. The part of the research is based on work done by myself jointly with Volodymyr Budzan and Mykola Mykhailych. Specifically, Mykola Mykhailych created the algorithm for synthetic motion blur and Volodymyr Budzan helped with the figures and experimental evaluation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Motion blur is one of the biggest and most unpleasant problems of photographers. The photographs can be degraded by blur both because of the shake of the camera and object movement. Moreover, the variation of a depth, blur radius, shape and other artifacts present on blurred photos make this problem even more complicated. This work is focused on removing motion blur from a single photograph, which is a specific case of an image-to-image translation. This highly challenging problem receives a lot of attention because of its importance for the the photographers and also because of a wide variety of applications, as deblurring an image allows to recover otherwise lost information. Thus, there is a wide variety of approaches to deblurring problem each of which has its own pros and cons. In general, the blur model is formulated as following: The common formulation of blur model is following:

$$I_B = K * I_S + N \tag{1.1}$$

, where I_B is a blurred image, K is a blur kernel, I_S is a sharp latent image, $*$ denotes the convolution operation and N is an additive noise. *et al.*

The whole family of deblurring methods is divided into two types: blind and non-blind deblurring. Early works [34] mostly focuses on non-blind deblurring, making an assumption that the blur function K is known. Most of them rely on the classical Lucy-Richardson algorithm, Wiener or Tikhonov filter to perform the deconvolution operation and obtain I_S estimate. In practice the blur function is unknown, so the blind deblurring algorithms estimate both latent sharp image I_S and blur kernel K. Finding a blur function for each pixel is an ill-posed problem, so most of the existing algorithms rely on image heuristics and assumptions on the sources of the blur. Those family of methods addresses the blur caused by camera shake by considering blur to be uniform across the image. Firstly, the camera motion is estimated in terms of the induced blur kernel, and then the effect is reversed by performing a deconvolution operation. Starting with the success of Fergus *et al.*[7], a lot derivative methods [39][38][26][2] has been developed over the last ten years. Some of the methods are based on an iterative approach [7] [39], which improve the estimate of the motion kernel and sharp image on each iteration by using parametric prior models. However, the running time, as well as the stopping criterion, is a significant problem for those kinds of algorithms. Others use assumptions of a local linearity of a blur function and simple heuristics to quickly estimate the unknown kernel. These methods are fast but work well on a small subset of images. With the success of deep learning, over the last few years, there appeared some approaches based on convolutional neural networks (CNNs). Sun *et al.* [33] use CNN to estimate blur kernel, Chakrabarti [5] predicts complex Fourier coefficients of motion kernel to perform non-blind deblurring in Fourier space whereas Gong [8] use fully convolutional network to move for motion flow estimation. All of these approaches use CNN to estimate the unknown blur function. Recently, a kernel-free end-to-end approaches by Noorozi [25] and Nah [23] that uses multi-scale CNN to directly deblur the image. Such methods are able to deal with different sources of the blur. The drawback is that the number of parameters in Multi-scale architectures raises problems with the inference time and a dataset for training, as very Deep Convolutional Neural Networks need thousands and millions of examples to be able to generalize well. Inspired by recent works on image super-resolution [18] and image-to-image translation by generative adversarial networks [13], we treat deblurring as a special case of such image-to-image translation. We present DeblurGAN – an approach based on Conditional Generative Adversarial Networks [22] and a multi-component loss function. Different from previous works we use Wasserstein GAN [1] with Gradient Penalty [10] and perceptual loss [14]. This encourages solutions which are perceptually hard to distinguish from real sharp images and allows to restore finer texture details in contrast to using traditional MSE or MAE as an optimization target.

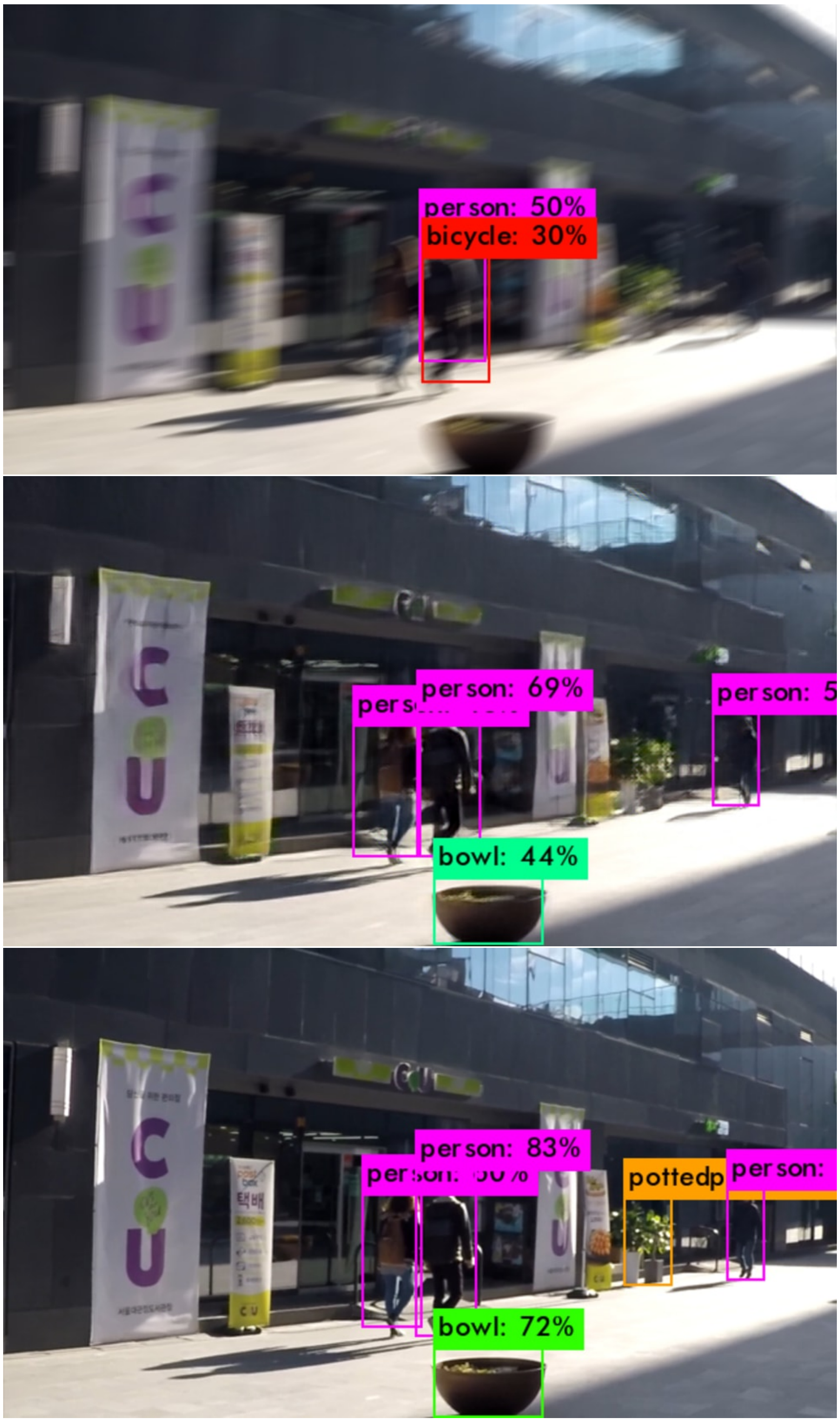


FIGURE 1.1: DeblurGAN helps object detection. From top to bottom: YOLO [28] detections results on a blurred image, restored by DeblurGAN, ground truth sharp image.

1.1 Contribution

In general, GAN are known for an ability to preserve high texture details in images and create solutions that are close to real image manifold and looks perceptually more convincing.
In our work we use ResNet with a global skip connection architecture and conditional GANs to form a perceptual loss function for non-uniform motion deblurring. Our main contributions are:

- We propose DeblurGAN, which is a cGAN-based network. In contrast to pure MSE-based solutions, all of the parts of our networks works towards creating sharp generated images with clear details. We use a combination of Wasserstein GAN [1] to move generated images towards sharp images manifold, VGG perceptual loss [14] to obtain "visually pleasing results", PatchGAN [4] [13] to approximate texture loss, Instance Normalization [35] which also imrpoves visual results and global residual connections which empirically shows huge boost in networks performance. All together, this architecture allows to set a new state of the art in blind motion deblurring.

- We propose a new way to generate synthetic realistic blurred images, based on random trajectories which helps to overcome the problem of training data limitation. We also construct new dataset which combines image pairs taken by high frame-rate camera in the wild and synthetically generated images and show training results of models trained on different combinations. We show that this combination allows to deal with blur caused both by camera shake and object movement.

- We confirm on different benchmarks that DeblurGAN is the new state of the art by measuring PSNR and SSIM metrics. Moreover, as PSNR metric captures presence of high texture details very poorly, we propose a new way to measure the quality of deblurring model based on results of object detection model (YOLO [28]). We refer to the new dataset and benchmark as DeblurTest.

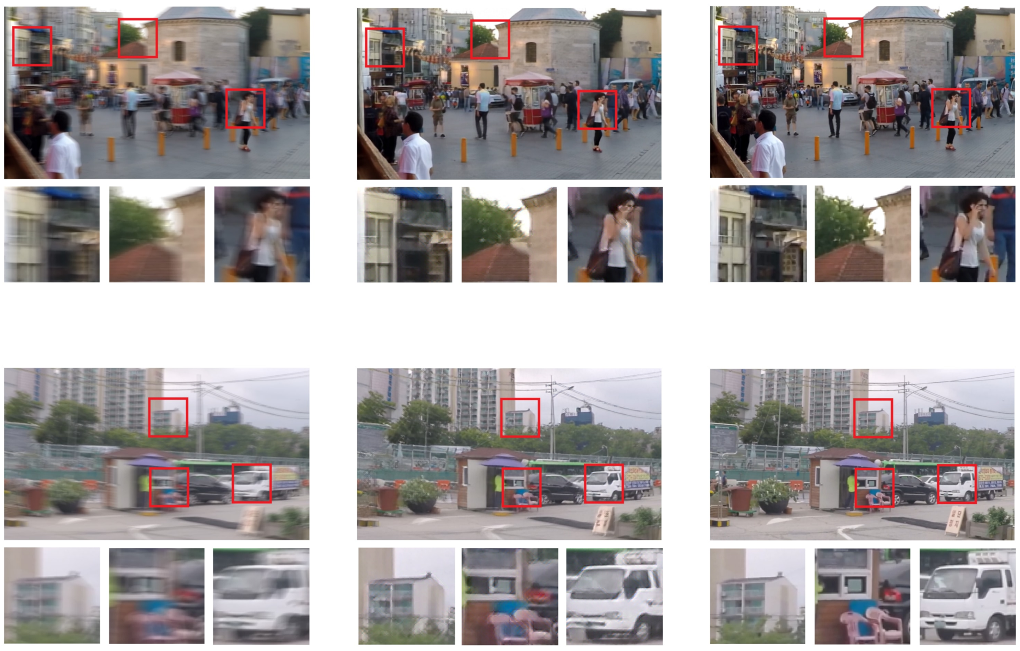


FIGURE 1.2: Images processed by DeblurGAN. From left to right: blurred photo, result of DeblurGAN, ground truth sharp image.

Chapter 2

Related work

2.1 Artificial Neural Networks

Neural network is a system of hardware and/or software created to mimic the work of neurons in the human brain. Neural networks, also called artificial neural networks, are a variety of machine learning technologies. The commercial application of these technologies mainly focuses on solving complex signal processing or pattern recognition problems. Examples of the commercial applications include handwriting recognition for check processing, speech to text, weather forecasting, and face recognition.
Neural networks typically include a large number of processors that run in parallel and arranged in layers. The first level receives initial information (input) - an analogue of optical nerves in the human visual processing. Each successive level receives an output from the previous level, and not from the input - in the same way, the neurons that are located far from the optic nerve receive signals from the closest one. The last layer produces system output.
Each processing node has its own small area of "expertise", including what it saw, and any rules that it originally programmed or developed for itself. The levels are highly interconnected, which means that each node of level n will be connected to many nodes of the level $n - 1$ - its inputs - and the level $n + 1$, which provides input for these nodes. The output layer may have one or more nodes, from which the answer that it creates can be read.
Neural networks are known for being adaptive, which means they modify themselves as they learn during the training, and the subsequent runs provide more information about the world. The simplest training model focuses on weighing input streams, so each node puts the importance of input from each of its predecessors. Inputs that that contribute

to getting the correct answers are weighted higher.
Typically, a neural network is initially trained on the large volumes of data. The data usually consists on pairs of input and ground truth output. For example, to build a network to identify actors, the dataset may include a series of photographs of actors, non-actors, masks, figures, animals, etc. Each entry is accompanied by an appropriate label, such as the names of actors, "not actor" or "non-human" information. Providing answers allows the model to adjust its internal weights to learn how to do its job better.

2.2 Convolutional Neural Networks

Convolutional neural networks [17] are a biologically inspired class of deep learning models that replace all the stages with one neural network, which studies in end-to-end manner, from row pixels to classifier outputs. Spatial structure of the images is explicitly used for regularization due to a limited connection between the layers (local filters), the sharing of parameters (convolutions) and special local invariance-building neurons (pooling layers). So, these architectures effectively shifted the necessary engineering from design of handcrafted image features to the network connection structure and hyperparameter tuning. Because of the computing limitations, until recently, CNN have been applied to a relatively small scale of the image problems, but improvements on hardware GPUs allowed CNNs to scale up to the millions of parameters, which in turn led to significant improvement of the classification of images, detection of objects [11] and others. In addition, features acquired by deep networks trained on ImageNet [6], have been shown to yield state-of-the-art performance across many standard image recognition datasets when classified with an SVM, even with no fine-tuning.
The main difference from the ordinary neural networks (multilayer perceptron) is that CNN architectures make an assumption that the inout is an image which allows us to encode certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network. The main three types of layers to build Convolutional Neural Network architectures [17][11] are Convolutional Layer, Pooling Layer, and Fully-Connected Layer. The layers are stacked together with a

different possible choices of the non-linearity to form a full architecture.

2.3 Generative adversarial networks

The idea of generative adversarial networks, introduced by Goodfellow *et al.*[9], is to define a game between two competing networks: discriminator and generator. The generator receives noise as an input and generates a sample. A discriminator receives a real and generated sample and is trying to distinguish between them. The goal of the generator is to fool the discriminator by generating perceptually convincing samples that can not be distinguished from the real one. From the theoretical perspective, the game between the generator G and discriminator D is the minimax objective:

$$\underset{G}{min}\,\underset{D}{max}\,\underset{x \backsim \mathbb{P}_r}{\mathbb{E}}[\log(D(x))] - \underset{\tilde{x} \backsim \mathbb{P}_g}{\mathbb{E}}[\log(1 - D(\tilde{x}))]$$

where $\mathbb{P}_r$ is the data distribution and $\mathbb{P}_g$ is the model distribution, defined by $\tilde{x} = G(z), z \backsim P(z)$, the input z is a sample from a simple noise distribution. GANs are known for its ability to generate samples of good perceptual quality, however, training of vanilla version suffer from many problems such as mode collapse, vanishing gradients etc, as described in [30]. Minimizing the value function in GAN is equal to minimizing the Jensen-Shannon divergence between the data and model distributions on x. Arjovsky *et al.* [1] discuss the difficulties in GAN training caused by JS divergence approximation and propose to use the Earth-Mover (also called Wasserstein-1) distance $W(q, p)$. The value function for WGAN is constructed using Kantorovich-Rubinstein duality [36]:

$$\underset{G}{min}\,\underset{D \in \mathcal{D}}{max}\,\underset{x \backsim \mathbb{P}_r}{\mathbb{E}}[D(x)] - \underset{\tilde{x} \backsim \mathbb{P}_g}{\mathbb{E}}[D(\tilde{x})] \tag{2.1}$$

where $\mathcal{D}$ is the set of $1-$Lipschitz functions and $\mathbb{P}_g$ is once again the model distribution The idea here is that critic value approximates $K \cdot W(P_r, P_\theta)$, where K is a Lipschitz constant and $W(P_r, P_\theta)$ is a Wasserstein distance. In this setting, a discriminator network is called critic and it approximates the distance between the samples. To enforce

Lipschitz constraint in WGAN Arjovsky *et al.*add weight clipping to $[-c, c]$. This technique, however, may lead to optimization difficulties such as capacity under-use or exploding/vanishing gradients. Gulrajani *et al.* [10] propose to add a gradient penalty term:

$$\lambda \underset{\tilde{x} \backsim \mathbb{P}_{\tilde{x}}}{\mathbb{E}} [(\|\nabla_{\tilde{x}} D(\tilde{x})\|_2 - 1)^2]$$

to the value function as an alternative way to enforce the Lipschitz constraint. This enables stable training of a wide variety of GAN architectures with almost no hyperparameter tuning.

2.4 Conditional adversarial networks

Generative Adversarial Networks have been applied to different image-to-image translation problems, such as super resolution [18], style transfer [20], product photo generation [4] and others. Isola *et al.* [13] provides a detailed overview of those approaches and present conditional GAN architecture also known as *pix2pix*. Unlike vanilla GAN, cGAN learns a mapping from observed image x and random noise vector z, to $y : G : x, z \rightarrow y$. Isola *et al.*also put a condition on the discriminator and use U-net architecture [29] for generator and Markovian discriminator which allows achieving perceptually superior results on many tasks, including synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images.

Chapter 3

Proposed Method

The goal is to recover sharp image I_S given only a blurred image I_B as an input, so no information about the blur kernel is provided. To do that we train a CNN G_{θ_G}, to which we refer as the Generator. For each I_B it estimates corresponding I_S image. In addition, during the training phase, we introduce critic function D_{θ_D} and train both networks in an adversarial manner.

3.1 Loss function

Designing appropriate loss function is crucial for training deep neural network, especially in image-to-image translation. Existing end to end learning approaches in blind motion deblurring are mostly using solely MSE as an optimization target [25]. The negative side of this choice is that MSE-based solution appears overly smooth due to the pixel-wise average of possible solutions in the pixel space, as shown in [18]. This results in often blurry recovered images without high texture details which is very important to avoid in image deblurring.
We formulate the loss function as a combination of content and adversarial loss:

$$\mathcal{L} = \underbrace{\underbrace{\mathcal{L}_{GAN}}_{adv\ loss} + \underbrace{\lambda \cdot \mathcal{L}_X}_{content\ loss}}_{total\ loss}$$

where the λ equals to 100 in all experiments. Unlike Isola *et al.*[13] we do not condition the discriminator as we do not need to penalize mismatch between the input and output. In our settings both of the loss components are needed to create perceptually convincing results, as generator goal is not only to create indistinguishable from natural sharp image manifold for critic but also to make sure that samples lie

close in L1, L2 or feature space. Below we would compare different possible choices for both adversarial and content loss.
Adversarial loss Most of the papers related to conditional GANs, use vanilla GAN objective as a loss [18][23]. More recently [41] provides an alternative way of using Least Square GAN [21] which is more stable and generates higher quality results. We use WGAN-GP [10] as the critic function. The critic does not output a probability that the reconstructed image is sharp and the loss is calculated as the following:

$$\mathcal{L}_{GAN} = \sum_{n=1}^{N} -D_{\theta_D}(G_{\theta_G}(I^B)) \tag{3.1}$$

Content loss Two classical choices for "content" loss function are **L1** or **MAE** loss, **L2** or **MSE** loss on raw pixels. Isola *et al.* [13] show in their work that L1 loss usually results in higher quality results with less blurring effect. Ledig *et al.* [18]and achieve state of the art in Super Resolution and Style Transfer using perceptual loss, which corresponds to visual similarity and helps to preserve high texture details. We build on top of their ideas and see a significant improvement in model results using the perceptual loss as the content loss part. Thus we define the content loss function as the following:

$$\mathcal{L}_X = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^S)_{x,y} - \phi_{i,j}(G_{\theta_G}(I^B))_{x,y})^2$$

where $\phi_{i,j}$ is the feature map obtained by the j-th convolution (after activation) before the i-th maxpooling layer within the VGG19 network, pretrained on ImageNet [6], $W_{i,j}$ and $H_{i,j}$ are the dimensions of the feature maps.

3.2 Network architecture

Generator CNN architecture is shown in Figure 3.1. It is similar to one proposed by Johnson *et al.*[14] for style transfer task. It contains two strided convolution blocks with stride $\frac{1}{2}$, nine residual blocks [11] (ResBlocks) and two transposed convolution blocks. Each ResBlock consists of a convolution layer, instance normalization layer [35], and ReLU [24] activation. Dropout [32] regularization with a probability of 0.5 is added after the first convolution layer in each ResBlock.

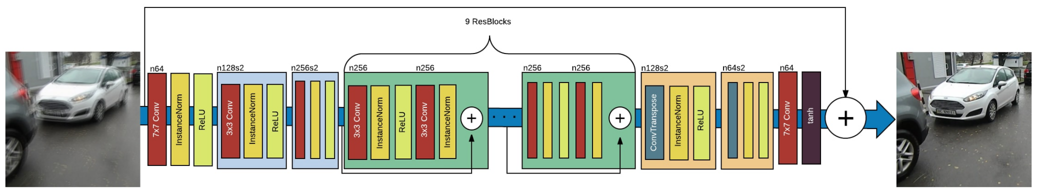


FIGURE 3.1: Architecture of the DeblurGAN generator network. DeblurGAN contains two strided convolution blocks with stride $\frac{1}{2}$, nine residual blocks [11] and two transposed convolution blocks. Each ResBlock consists of a convolution layer, instance normalization layer, and ReLU activation.

In addition, we introduce the global skip connection which we refer to as ResOut. CNN learns a residual correction I_R to the blurred image I_B, so $I_S = I_B + I_R$. We find that such formulation makes training faster and resulting model generalizes better.

During the training phase, we define a critic network D_{θ_D}, which is Wasserstein GAN [1] with Gradient Penalty [10], to which we refer as WGAN-GP. The architecture of critic network is identical to PatchGAN [13, 20] that penalizes structures at the scale of patches and can be interpreted as Texture/Style loss approximation. All the convolutional layers except the last are followed by InstanceNorm layer and LeakyReLU [37] with $\alpha = 0.2$.

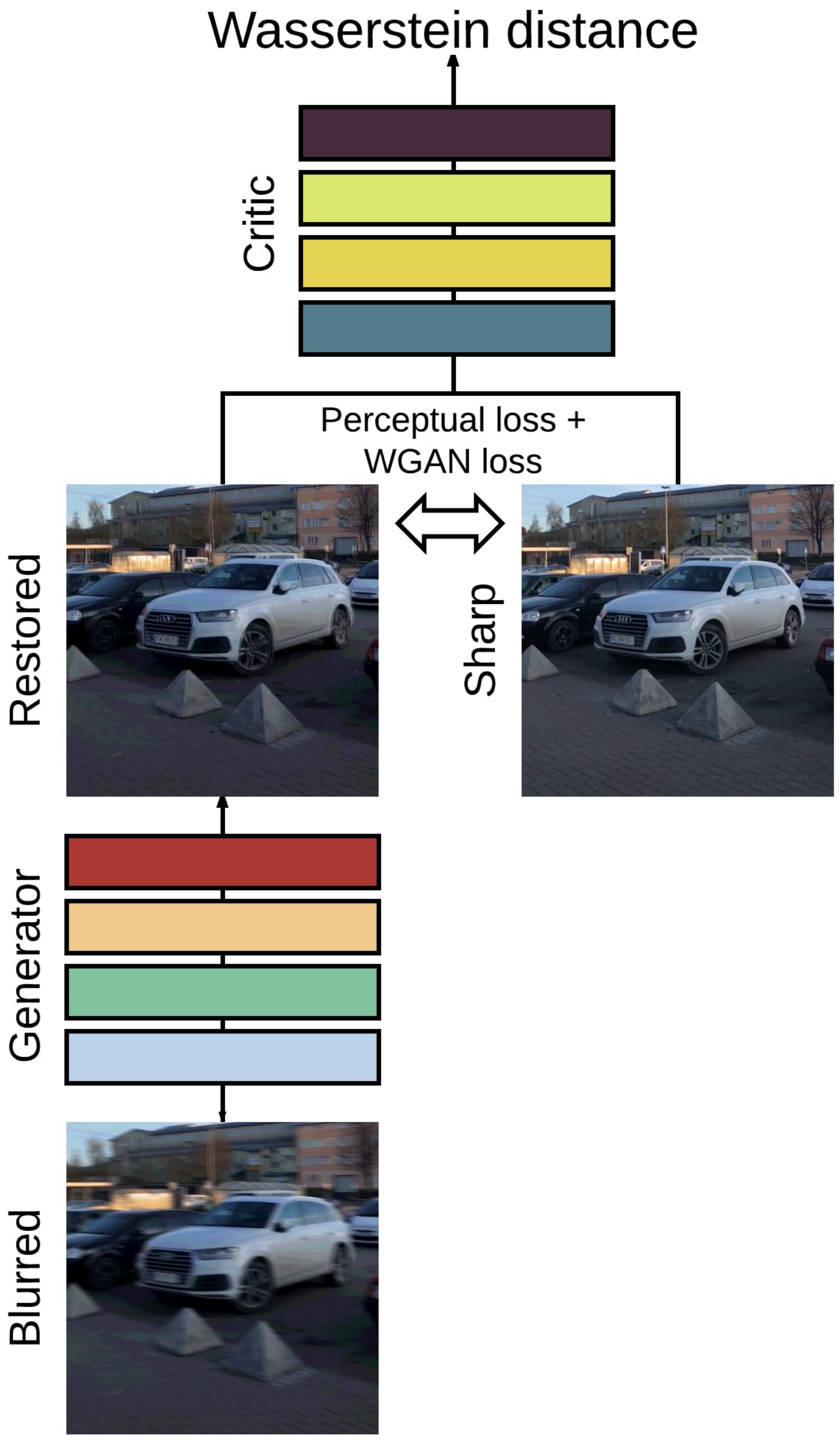


FIGURE 3.2: Conditional GAN for motion deblurring. Generator network takes the blurred image as an input and produces the estimate of the sharp image. During the training time, the critic network takes restored and sharp image as an input and estimates a distance between them. Total loss consists of WGAN loss from critic and perceptual loss [14] based on the difference in activations of VGG-19 [31] between the feature maps of sharp and restored images. At test time only the generator is kept

3.3 Motion blur generation

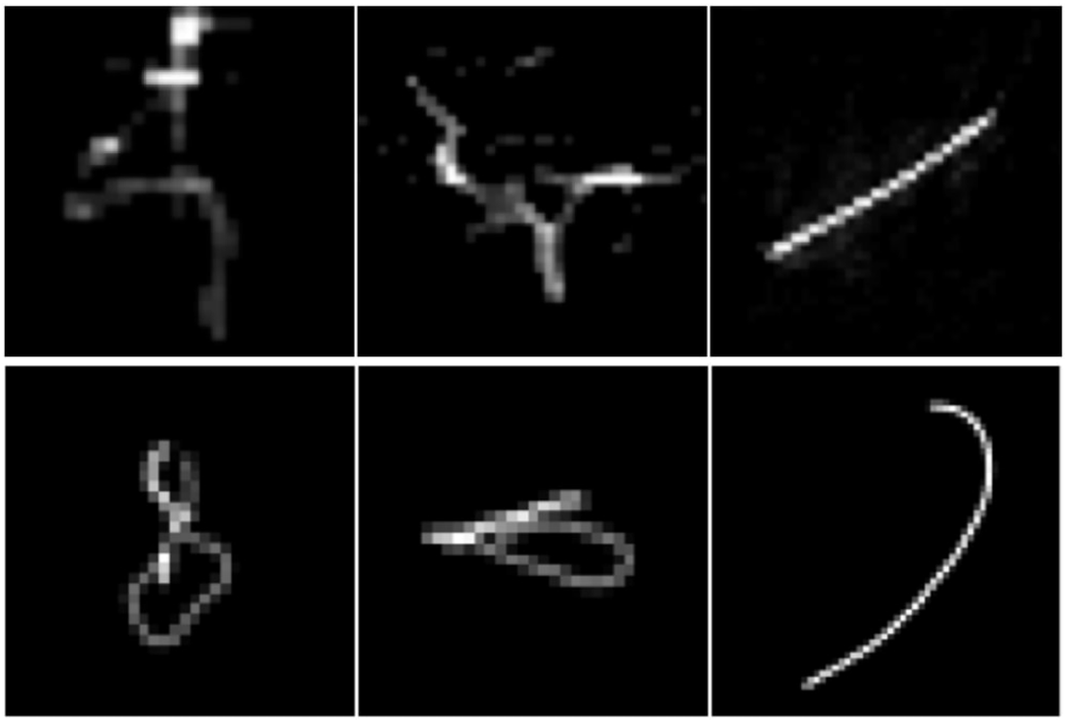


FIGURE 3.3: Top row: Blur kernels from real-world images estimated by Fergus *et al.*[7]. Bottom row: Synthetically generated kernels by our method. Our randomized method can simulate wide variety of realistic blur kernels with different level of non-linearity.

There is no easy method to obtain image pairs of corresponding sharp and blurred images for training, in contrast to other popular image-to-image translation problems, such as super-resolution or colorization. A typical approach to obtain image pairs for training is to use a high frame-rate camera to simulate blur using average of sharp frames from video [25, 23]. It allows to create realistic blurred images but limits the image space only to scenes present in taken videos and makes it complicated to scale the dataset. Sun *et al.*[33] creates synthetically blurred images by convolving clean natural images with one out of 73 possible linear motion kernels, Xu *et al.*[40] also use linear motion kernels to create synthetically blurred images. Chakrabarti [5] creates blur kernel by sampling 6 random points and fitting a spline to them.

We take a step further and propose a method, which simulates more realistic and complex blur kernels. We follow the idea described by

Boracchi and Foi [3] of random trajectories generation. Then the kernels are generated by applying sub-pixel interpolation to the trajectory vector. Each trajectory vector is a complex valued vector, which corresponds to the discrete positions of an object following 2D random motion in a continuous domain. Trajectory generation is done by Markov process, summarized in Algorithm 1. Position of the next point of the trajectory is randomly generated based on the previous point velocity and position, gaussian perturbation, impulse perturbation and deterministic inertial component

Algorithm 1 Motion blur kernel generation. Initial parameters:
$M = 2000$ – number of iterations, $L_{max} = 60$ – max length of the movement,
$p_s = 0.001$ – probability of impulsive shake.
I = inertia term, uniform from (0,0.7)
p_b – probability of big shake, uniform from (0,0.2)
p_g – probability of gaussian shake, uniform from (0,0.7)
ϕ – initial angle, uniform from (0,2π)
x – trajectory vector

```
1:  procedure BLUR(Img, $M, L_{max}, p_s$)
2:      $v_0 \leftarrow cos(\phi) + sin(\phi) * i$
3:      $v \leftarrow v_o * L_{max}/(M-1)$
4:      $x = \text{zeros}(M, 1)$
5:      for $t = 1$ to $M-1$ do
6:          if randn $< p_b * p_s$ then
7:              nextDir $\leftarrow 2 \cdot v \cdot e^{i*(\pi+(\text{randn}-0.5))})$
8:          else:
9:              nextDir $\leftarrow 0$
10:         $dv \leftarrow \text{nextDir} + p_s * (p_g * (\text{randn} + i * \text{randn}) * I * x[t] * (L_{max}/(M-1))$
11:         $v \leftarrow v + dv$
12:         $v \leftarrow (v/abs(v)) * L_{max}/(M-1)$
13:         $x[t+1] \leftarrow x[t] + v$
14:     Kernel $\leftarrow$ sub pixel interpolation$(x)$
15:     Blurred image $\leftarrow conv(\text{Kernel}, \text{Img})$
16:     return Blurred image
```

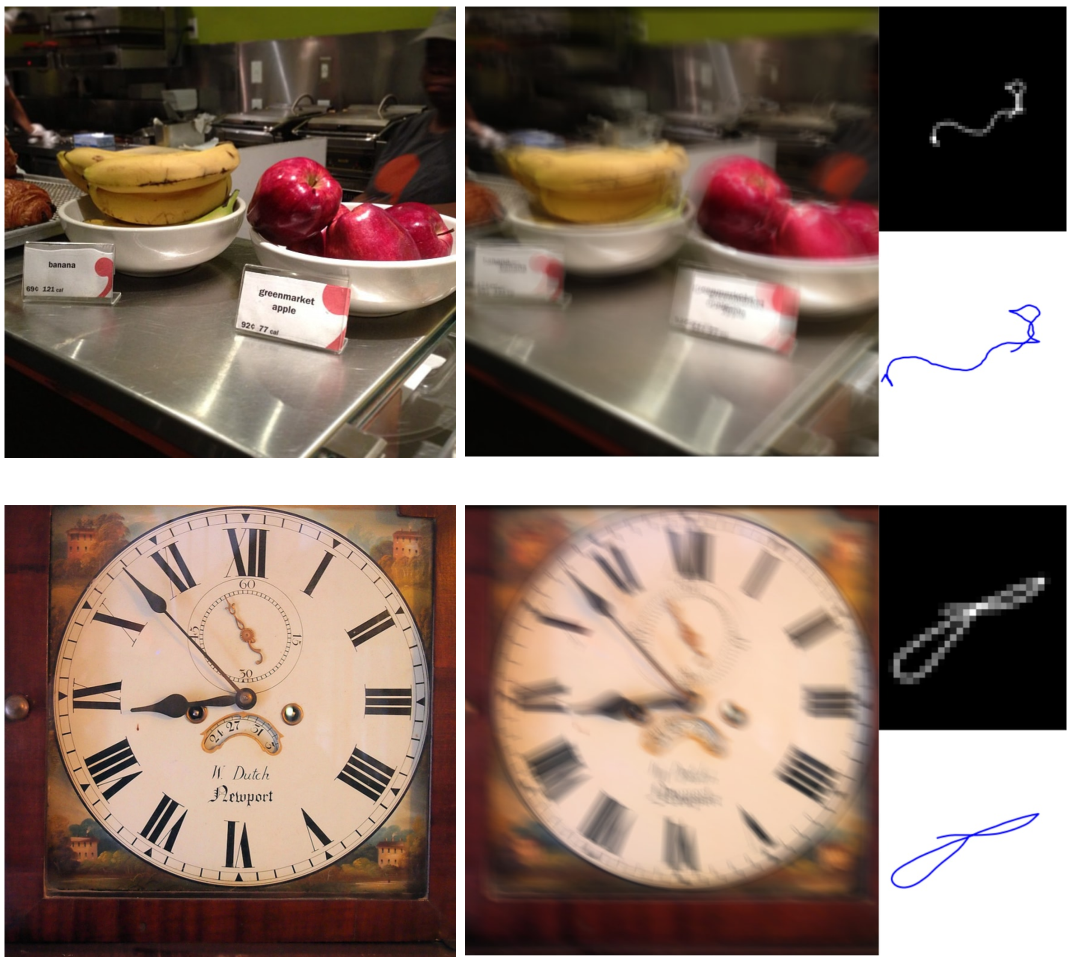


FIGURE 3.4: Examples of generated camera motion trajectory and the blur kernel and the corresponding blurred images.

3.4 Training Details

We implemented all of our models using PyTorch[27] deep learning framework. The training was performed on a single Titan-X GPU using three different datasets. The first model to which we refer as *DeblurGAN$_{WILD}$* was trained on a random crops of size 256x256 from GoPro dataset [23]. The second one *DeblurGAN$_{Synth}$* was trained on 256x256 patches from MS COCO dataset blurred by method, presented in previous Section. We also trained *DeblurGAN$_{Comb}$* on a combination of synthetically blurred images and images taken in the wild, where the ratio of synthetically generated images to the images taken by a high frame-rate camera is 2:1. As the models are fully convolutional and are trained on image patches they can be applied to images of arbitrary size. For optimization we follow the approach of [1] and perform 5 gradient descent steps on D_{θ_D}, then one step on G_{θ_G}, using Adam [15] as a solver. The learning rate is set initially to 10^{-4} for both generator and critic. After the first 150 epochs we linearly decay the rate to zero over the next 150 epochs. At inference time we follow the idea of [13] and apply both dropout and instance normalization. All the models were trained with a batch size = 1, which showed empirically better results on validation. The training phase took 6 days for training one *DeblurGAN* network. We see that DeblurGAN trained only on synthetic data still has poorer generalization than other models and is not able to restore images with highly non-uniform blur and blur caused by object movement. However, a combination of synthetically generated images with images taken by a high-frame rate camera increase network generalization abilities. Still, the $DeblurGAN_{WILD}$ produces the smoothest results as synthetic blurred images have different blur radius, so some results of $DeblurGAN_{Comb}$ has visible ringing artifacts. The performance of all of our models are presented in the next chapter.

Chapter 4

Experimental evaluation

The evaluation of image-to-image translation tasks is a difficult problem [13] [30]. As already mentioned, often perceptually more convincing generated images have lower score on classical metrics such as PSNR. In our work, we propose a new way to evaluate the quality of deblurring model on a real-world problem and show the results of our models on the new benchmark as well as on two different widely-used benchmarks. We show on those benchmarks that*DeblurGAN*$_{WILD}$ is a new state of the art for blind motion deblurring.

4.1 GoPro Dataset

TABLE 4.1: Mean peak signal-to-noise ratio and structural similarity measure on GoPro test dataset of 1111 images. We took *linear* image subset for testing all models. State-of-art results $(*)$ by Nah *et al.* [23] obtained with *gamma* subset.

Method	Sun *et al.*	Nah *et al.*	DeblurGAN		
Metric	[33]	[23]	*WILD*	*Synth*	*Comb*
PSNR	24.64	28.34/**29.1***	27.21	23.69	**28.72**
SSIM	0.842	0.916	0.954	0.884	**0.958**
Time	20 min	4.33 s	**0.85 s**		

GoPro dataset[23] consists of 2103 pairs of blurred and sharp images in 720p quality, taken from various scenes. We compare the results of our models with state of the art models [33], [23] on standard metrics and also show the running time of each algorithm on a single **GPU**. Results are in Table4.1. DeblurGAN shows superior results in terms of structured self-similarity, is close to state-of-the-art in peak signal-to-noise-ratio and provides better looking results by visual inspection. It can handle blur caused by camera shake and object movement, does not suffer from usual artifacts in kernel estimation methods and at the same time has more than 6x fewer parameters comparing to Multi-scale CNN , which heavily speeds up the inference. Deblured images from test on GoPro dataset are shown in Figure 4.1.

4.2 Kohler dataset

Kohler dataset [16] consists of 4 images blurred with 12 different kernels for each of them. This is a standard benchmark dataset for evaluation of blind deblurring algorithms. The dataset is generated by recording and analyzing real camera motion, which is played back on a robot platform such that a sequence of sharp images is recorded sampling the 6D camera motion trajectory. Results are in Table 4.2

TABLE 4.2: Mean peak signal-to-noise ratio and structural similarity measure on Kohler benchmark dataset.

Method	Sun *et al.*	Nah *et al.*	DeblurGAN		
Metric	[33]	[23]	*WILD*	*Synth*	*Comb*
PSNR	25.22	26.02	**26.10**	25.67	25.86
MSSIM	0.773	0.811	**0.816**	0.792	0.802

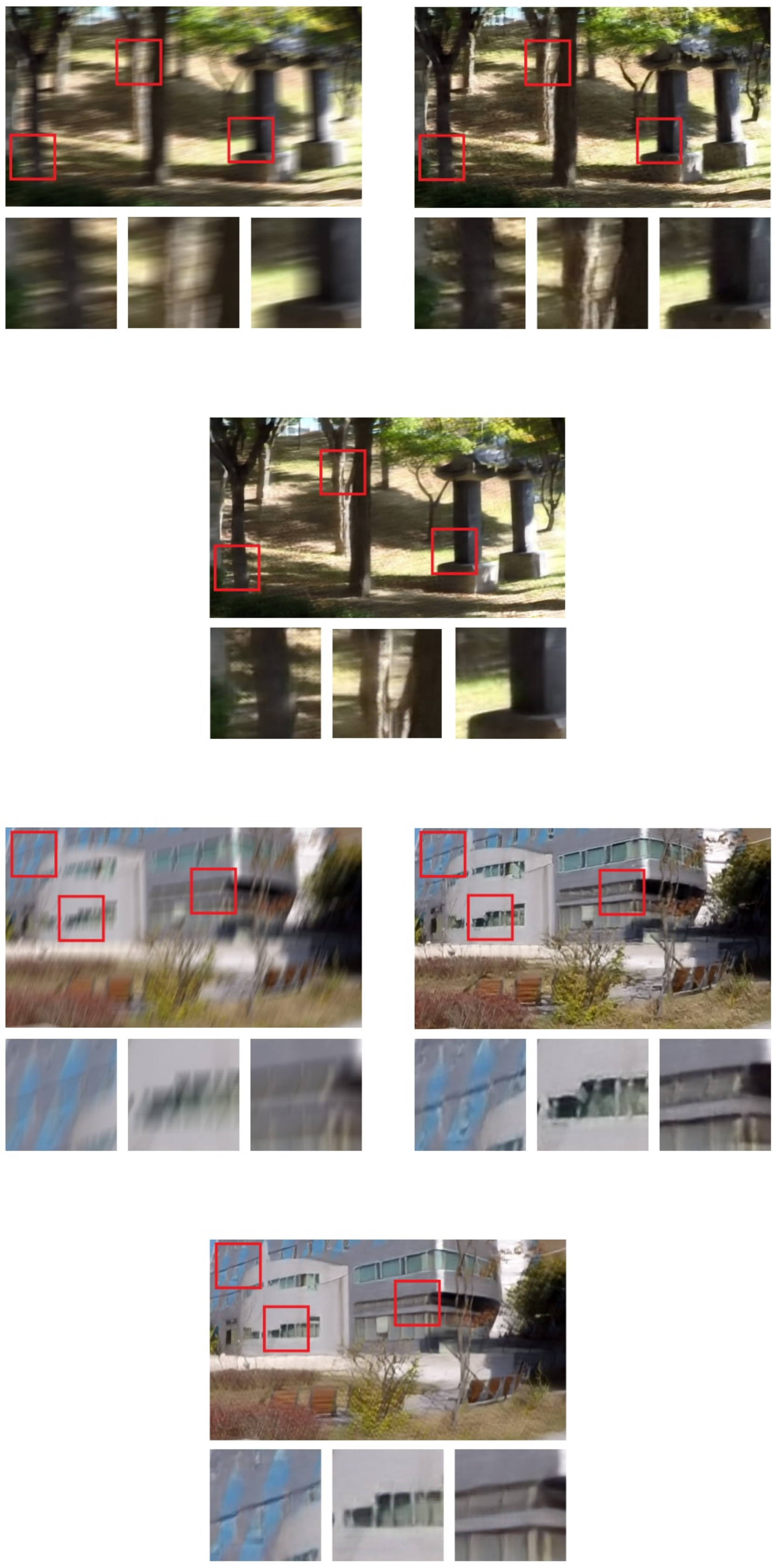


FIGURE 4.1: Results of evaluation on GoPro test dataset. From left to right: blurred photo, result of Nah *et al.* [23], result of our algorithm.

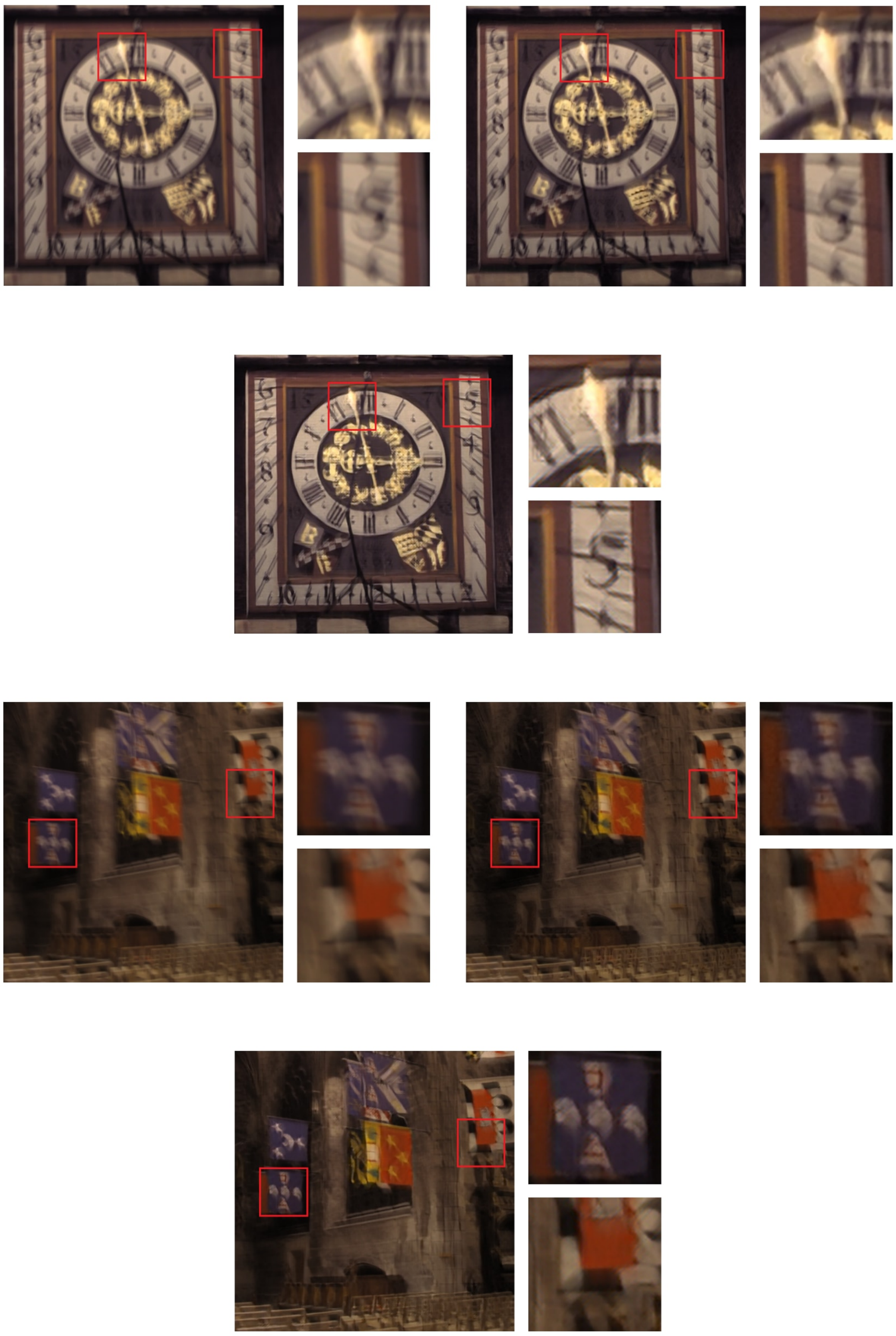
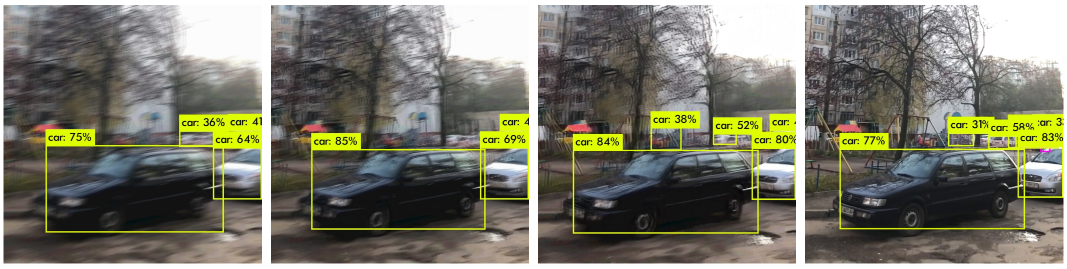


FIGURE 4.2: Evaluation on Kohler dataset. From left to right: blurred photo, Nah *et al.*[23], DeblurGAN

4.3 Object Detection benchmark on YOLO



(A) Blurred photo (B) Nah *et al.* [23] (C) DeblurGAN (D) Sharp photo

FIGURE 4.3: YOLO object detection before and after deblurring

Object Detection is one of the most well-studied problems in computer vision with applications in different domains from autonomous driving to security. During the last few years approaches based on Deep Convolutional Neural Networks showed state of the art performance comparing to traditional methods. However, those networks are trained on limited datasets and in real-world settings images are often degraded by different artifacts, including motion blur, Similar to [19] we studied the influence of motion blur on object detection and propose a new way to evaluate the quality of deblurring algorithm based on results of object detection on a pretrained YOLO [28] network.

For this, we constructed a dataset of sharp and blurred street views by simulating camera shake using a high frame-rate video camera. Following [12][23][25] we take a random between 5 and 25 frames taken by 240fps camera and compute the blurred version of a middle frame as an average of those frames. All the frames are gamma-corrected with $\gamma = 2.2$ and then the inverse function is taken to obtain the final blurred frame. Overall, the dataset consists of 410 pairs of blurred and sharp images, taken from the streets and parking places with different number and types of cars.

Blur source includes both camera shake and blur caused by car movement. The dataset and supplementary code are available online. Then sharp images are feed into the YOLO network and the result after visual verification is assigned as ground truth. Then YOLO is run on blurred and recovered versions of images and average recall and precision between obtained results and ground truth are calculated. This approach corresponds to the quality of deblurring models on real-life problems and correlates with the visual quality and sharpness of the generated images, in contrast to standard PSNR metric. The precision, in general, is higher on blurry images as there are no sharp object boundaries and smaller object are not detected as it shown in Figure 4.3.

Results are shown in Table 4.3. DeblurGAN significantly outperforms competitors in terms of recall and F1 score.

TABLE 4.3: Results of object detection on YOLO [28] on blurred and restored photos using our and Nah *et al.* [23] algorithms. We take the results on corresponding sharp images as the ground truth. DeblurGAN has higher recall and F1 score than its competitors.

Method	prec.	recall	F1 score
no deblur	0.821	0.437	0.570
Nah *et al.* [23]	**0.834**	0.552	0.665
DeblurGAN WILD	0.764	0.631	0.691
DeblurGAN synth	0.801	0.517	0.628
DeblurGAN comb	0.671	**0.742**	**0.704**

Chapter 5

Conclusion

We described a kernel-free blind motion deblurring learning approach and introduced DeblurGAN which is a Conditional Adversarial Network that is optimized using a multi-component loss function. In addition to this, we implemented a new method for creating a realistic synthetic motion blur able to model different blur sources. We introduce a new benchmark and evaluation protocol based on results of object detection.

Bibliography

Arjovsky, M., S. Chintala, and L. Bottou (2017). “Wasserstein GAN”. In: *ArXiv e-prints*. arXiv: 1701.07875 [stat.ML].

Babacan, S. D. et al. (2012). “Bayesian Blind Deconvolution With General Sparse Image Priors”. In: *European Conference on Computer Vision (ECCV)*. Firenze, Italy: Springer.

Boracchi, G. and A. Foi (2012). “Modeling the Performance of Image Restoration From Motion Blur”. In: *Image Processing, IEEE Transactions on* 21.8, pp. 3502 –3517. ISSN: 1057-7149. DOI: 10.1109/TIP.2012.2192126.

Bousmalis, K. et al. (2016). “Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks”. In: *ArXiv e-prints*. arXiv: 1612.05424 [cs.CV].

Chakrabarti, Ayan (2016). “A neural approach to blind motion deblurring”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319464862. DOI: 10.1007/978-3-319-46487-9{_}14.

Deng, J. et al. (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*.

Fergus, Rob et al. (2006). “Removing Camera Shake from a Single Photograph”. In: *ACM Trans. Graph.* 25.3, pp. 787–794. ISSN: 0730-0301. DOI: 10.1145/1141911.1141956. URL: http://doi.acm.org/10.1145/1141911.1141956.

Gong, Dong et al. “From Motion Blur to Motion Flow: a Deep Learning Solution for Removing Heterogeneous Motion Blur”. In:

Goodfellow, Ian J. et al. (2014). “Generative Adversarial Networks”. In: URL: https://arxiv.org/abs/1406.2661 (visited on 01/08/2017).

Gulrajani, I. et al. (2017). “Improved Training of Wasserstein GANs”. In: *ArXiv e-prints*. arXiv: 1704.00028 [cs.LG].

He, Kaiming et al. (2015). “Deep Residual Learning for Image Recognition”. In: *arXiv preprint arXiv:1512.03385*.

Hirsch, Michael et al. (2011). "Fast Removal of Non-uniform Camera Shake". In: *Proceedings of the 2011 International Conference on Computer Vision*. ICCV '11. Washington, DC, USA: IEEE Computer Society, pp. 463–470. ISBN: 978-1-4577-1101-5. DOI: 10.1109/ICCV.2011.6126276. URL: http://dx.doi.org/10.1109/ICCV.2011.6126276.

Isola, Phillip et al. (2016). "Image-to-Image Translation with Conditional Adversarial Networks". In: *arxiv*.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision*.

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization." In: *CoRR* abs/1412.6980. URL: http://dblp.uni-trier.de/db/journals/corr/corr1412.html#KingmaB14.

Köhler, Rolf et al. (2012). "Recording and Playback of Camera Shake: Benchmarking Blind Deconvolution with a Real-world Database". In: *Proceedings of the 12th European Conference on Computer Vision - Volume Part VII*. ECCV'12. Florence, Italy: Springer-Verlag, pp. 27–40. ISBN: 978-3-642-33785-7. DOI: 10.1007/978-3-642-33786-4_3. URL: http://dx.doi.org/10.1007/978-3-642-33786-4_3.

LeCun, Y. et al. (1992). "Handwritten digit recognition with a back-propagation network". In: *Neural Netwotks, current applications*. Ed. by Lisboa P.G.J. Chappman and Hall.

Ledig, C. et al. (2016). "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *ArXiv e-prints*. arXiv: 1609.04802 [cs.CV].

Li, B. et al. (2017). "An All-in-One Network for Dehazing and Beyond". In: *ArXiv e-prints*. arXiv: 1707.06543 [cs.CV].

Li, C. and M. Wand (2016). "Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks". In: *ArXiv e-prints*. arXiv: 1604.04382 [cs.CV].

Mao, Xudong et al. (2016). *Least Squares Generative Adversarial Networks*. cite arxiv:1611.04076. URL: http://arxiv.org/abs/1611.04076.

Mirza, Mehdi and Simon Osindero (2014). "Conditional Generative Adversarial Nets". In: *CoRR* abs/1411.1784. arXiv: 1411.1784. URL: http://arxiv.org/abs/1411.1784.

Nah, Seungjun et al. (2016). “Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring”. In:

Nair, Vinod and Geoffrey E. Hinton (2010). “Rectified linear units improve restricted boltzmann machines”. In: *International Conference on Machine Learning (ICML)*, pp. 807–814.

Noroozi, Mehdi, Paramanand Chandramouli, and Paolo Favaro (2017). “Motion Deblurring in the Wild”. In:

Perrone, Daniele and Paolo Favaro (2014). “Total Variation Blind Deconvolution: The Devil is in the Details”. In: *EEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

PyTorch. http://pytorch.org.

Redmon, J. et al. (2015). “You Only Look Once: Unified, Real-Time Object Detection”. In: *ArXiv e-prints*. arXiv: 1506.02640 [cs.CV].

Ronneberger, O., P. Fischer, and T. Brox (2015). “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *ArXiv e-prints*. arXiv: 1505.04597 [cs.CV].

Salimans, T. et al. (2016). “Improved Techniques for Training GANs”. In: *ArXiv e-prints*. arXiv: 1606.03498 [cs.LG].

Simonyan, K. and A. Zisserman (2014). “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *ArXiv e-prints*. arXiv: 1409.1556 [cs.CV].

Srivastava, Nitish et al. (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *J. Mach. Learn. Res.* 15.1, pp. 1929–1958. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=2627435.2670313.

Sun, Jian et al. “Learning a Convolutional Neural Network for Nonuniform Motion Blur Removal”. In:

Szeliski, Richard (2010). *Computer Vision: Algorithms and Applications*. 1st. New York, NY, USA: Springer-Verlag New York, Inc. ISBN: 1848829345, 9781848829343.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor S. Lempitsky (2016). “Instance Normalization: The Missing Ingredient for Fast Stylization”. In: *CoRR* abs/1607.08022. arXiv: 1607.08022. URL: http://arxiv.org/abs/1607.08022.

Villani, C. (2008). *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg. ISBN: 9783540710509. URL: https://books.google.com.ua/books?id=hV8o5R7_5tkC.

Xu, Bing et al. (2015). “Empirical evaluation of rectified activations in convolutional network”. In: *arXiv preprint arXiv:1505.00853*.

Xu, Li and Jiaya Jia (2010). “Two-phase kernel estimation for robust motion deblurring”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642155480. DOI: 10.1007/978-3-642-15549-9{_}12.

Xu, Li, Shicheng Zheng, and Jiaya Jia. “Unnatural L 0 Sparse Representation for Natural Image Deblurring”. In: URL: http://www.cse.cuhk.edu.hk/leojia/projects/l0deblur/.

Xu, Li et al. (2014). “Deep Convolutional Neural Network for Image Deconvolution”. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*. NIPS’14. Montreal, Canada: MIT Press, pp. 1790–1798. URL: http://dl.acm.org/citation.cfm?id=2968826.2969026.

Zhu, Jun-Yan et al. (2017). “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *arXiv preprint arXiv:1703.10593*.